

# Packet Latency of Deterministic Broadcasting in Adversarial Multiple Access Channels <sup>\*</sup>

Lakshmi Anantharamu <sup>†</sup>      Bogdan S. Chlebus <sup>†</sup>      Dariusz R. Kowalski <sup>‡</sup>  
Mariusz A. Rokicki <sup>‡</sup>

## Abstract

We study broadcasting on multiple access channels with dynamic packet arrivals and jamming. The communication environments is represented by adversarial models which specify constraints on packet arrivals and jamming. We consider deterministic distributed broadcast algorithms and give upper bounds on the worst-case packet latency and the number of queued packets in relation to the parameters defining adversaries. Packet arrivals are determined by the rate of injections and number of packets that can arrive in one round. Jamming is constrained by the rate with which the adversary can jam rounds and by the number of consecutive rounds that can be jammed.

**Keywords:** multiple access channel, adversarial queuing, jamming, distributed algorithm, deterministic algorithm, packet latency, queue size.

---

<sup>\*</sup>The results of this paper appeared in a preliminary form in [6] and [7]. The work of the first and second authors was supported by the NSF Grant 1016847.

<sup>†</sup>Department of Computer Science and Engineering, University of Colorado Denver, Denver, Colorado, U.S.A.

<sup>‡</sup>Department of Computer Science, University of Liverpool, Liverpool, U.K.

# 1 Introduction

We study broadcasting on multiple access channels by deterministic distributed algorithms. The communication medium is considered either with jamming or without it. We evaluate the performance of communication algorithms by bounds on packet latency and number of packets queued at stations, both these metrics understood in their worst-case sense.

The traditional approach to dynamic broadcasting on multiple access channels uses randomization to arbitrate for access to a channel in a distributed manner. Typical examples of randomized protocols include backoff protocols, like the binary exponential backoff employed in the Ethernet. It is a popular opinion that using randomization is the only option in order to be able to cope with bursty traffic, subject to challenges of collision resolution, and that effectiveness of deterministic solutions for dynamic broadcasting is limited.

The preliminary conference presentations of this work [6, 7] related outcomes of simulating experiments in which we measured average packet latency of various broadcasting algorithms, including deterministic and backoff ones. These experiments indicated that when injections are intense and sustained then even simple deterministic algorithms perform better than randomized backoff ones in terms of average packet latency. This indicates that deterministic algorithms for broadcasting on multiple-access channels have untapped potential.

We explore algorithmic paradigms useful for deterministic distributed broadcasting with dynamic continuous packet injection. This is done not that much in order to compare deterministic algorithms to randomized solutions but rather as an algorithmic problem interesting in its own sake. The goal is to investigate worst-case bounds on packet latency and queue size achievable by deterministic solutions to dynamic broadcasting. It is supported by a model of continuous packet injection without any stochastic assumptions about how packets are generated and where and when they are injected. This model of adversarial queueing is an alternative to models of stochastic packet injection. It has proved useful in studying dynamic communication with only minimal constraints on how traffic is generated.

There are numerous reasons why the traditional approach to broadcasting by employing randomization has been considered as essentially the only viable one. The methodological underpinnings of key performance metrics, like queue size and latency, have normally been studied with stochastic assumptions in mind. The methodology of simulations has been geared towards models of regular data injection defined by simple stochastic constraints. Most importantly, in real-world applications, most stations stay idle for most of the time, so that periods of inactivity are interspersed with unexpected bursts of activity by stations in unpredictable configurations. The success of Ethernet, as a real-world implementation of local area networks, has confirmed that randomization works very well in practice.

Jamming in wireless networks is often understood as an effect of deliberate transmissions of radio signals that disrupt the flow of information by creating interference of legitimate signals with such additional disrupting transmissions. We use the word “jamming” in a different meaning. A jammed round has the same effect as one with multiple simultaneous transmissions, in how it is perceived by the stations attached to the channel. Stations cannot distinguish a jammed round from a round with multiple transmissions. This means that jamming is understood as logical or virtual, in the sense that we do not make any assumptions about the physical reasons justifying a possibility that one station transmits in a round and the transmitted message is not heard on

the channel. This logical approach to jamming allows to capture a situation in which jamming occurs because groups of stations execute their independent communication protocols so that for each group an interference caused by “foreign” transmissions is logically equivalent to jamming. A similar motivation arrives from a situation in which a degradation-of-service attack produces dummy packets that interfere with legitimate packets.

We investigate deterministic broadcast algorithms for dynamic packet injection. No randomization in algorithms nor any stochastic component determining packet injection is present in the considered communication environments. The obtained upper bounds on packet latency and queue sizes of broadcast algorithms are worst-case.

The studied communication algorithms are distributed in that they are executed with no centralized control. The stations attached to the channel are assumed to be activated at the same initial round with empty queues.

We consider broadcasting against adversaries that control both injections of packets into stations and jamming of the communication medium. Packet injection is limited only by the rate of injecting new packets and the number of packets that can be injected simultaneously. Jamming is limited by the rate of jamming different rounds and by how many consecutive rounds can be jammed.

We use the slotted model of synchrony, in which an execution of a communication algorithm is partitioned into rounds, so that a transmission of a message with one packet takes one round. The set of stations attached to the channel is fixed and their number  $n$  is known, in that it can be used in codes of algorithms. Stations are equipped with private queues, in which they can store packets until they are transmitted successfully.

Synchrony is the underlying model’s component that allows to define the rate of injecting packets and the rate of jamming different rounds. Similarly, rounds are needed to determine burstiness of traffic, determined as the maximum number of packets that can be injected “at the same time,” and the burstiness of jamming, understood as the maximum size of a time interval that is unavailable for successful transmissions because of continuous jamming.

All the considered algorithms have bounded packet latency for each fixed injection rate  $\rho$  and jamming rate  $\lambda$  subject only to the necessary constraint that  $\rho + \lambda < 1$ . The obtained results are recapitulated in the final Section 7.

**Previous work on adversarial multiple access channels.** Now we review previous work on broadcasting in multiple-access channels in the framework of adversarial queuing. The first such work, by Bender et al. [15], concerned throughput of randomized backoff for multiple-access channels, in the queue-free model. Deterministic distributed broadcast algorithms for multiple-access channels, in the model of stations with queues, were first considered by Chlebus et al. [24]. They specified the classes of acknowledgment based and full sensing deterministic distributed algorithms, along the lines of the respective randomized protocols [21].

The maximum throughput, defined to mean the maximum rate for which stability is achievable, was studied by Chlebus et al. [23]. Their model was of a fixed set of stations with queues, whose size  $n$  is known. They developed a stable deterministic distributed broadcast algorithm with queues of  $\mathcal{O}(n^2 + \text{burstiness})$  size against leaky-bucket adversaries of injection rate 1, where “burstiness” is the maximum number of packets that the adversary can inject in one round. That work demonstrated that throughput 1 was achievable in the model of a fixed set of stations whose number  $n$  is known. They also showed some restrictions on traffic with throughput 1; in particular,

communication algorithms have to be adaptive (use control bits in messages), achieving bounded packet latency is impossible, and queues have to be  $\Omega(n^2 + \text{burstiness})$ .

Anantharamu et al. [8] extended work on throughput 1 in adversarial settings by studying the impact of limiting a window-type adversary by assigning individual rates of injecting data for each station. They gave a non-adaptive algorithm for channels without collision detection of  $\mathcal{O}(n + w)$  queue size and  $\mathcal{O}(nw)$  packet latency, where  $w$  is the window size; this is in contrast with general adversaries, against whom bounded packet latency for injection rate 1 is impossible to achieve.

Bieńkowski et al. [18] studied online broadcasting against adversaries that are unbounded in the sense that they can inject packets into arbitrary stations with no constraints on their numbers nor rates of injection. They gave a deterministic algorithm optimal with respect to competitive performance, when measuring either the total number of packets in the system or the maximum queue size. They also showed that the algorithm is stochastically optimal for any expected injection rate smaller than or equal to 1.

Anantharamu and Chlebus [5] considered a multiple access channel with an unbounded supply of anonymous stations attached to it, among which only stations activated by the adversary with injected packets participate in broadcasting. They studied deterministic distributed broadcast algorithms against adversaries that are restricted to be able to activate at most one station per round. Their algorithms can provide bounded packet latency for injection rates up to  $1/2$ , with specific rates depending on additional features of algorithms, and showed that no injection rate greater than  $\frac{3}{4}$  can be handled with bounded packet latency in this model.

**Related work.** The simplest communication problem on multiple access channels is about collision resolution, in which there is a group of active stations, being a subset of all stations connected to the channel, and we want to have either some station in the group or all of them transmit at least once successfully. For recent work on this topic, see the papers by Kowalski [32], Anta et al. [13], and De Marco and Kowalski [25].

Most related work on broadcasting on multiple access channels has been carried out with randomization playing an integral part. Randomness can affect the behavior of protocols either directly, by being a part of the mechanism of a communication algorithm, or indirectly, when packets are generated subject to stochastic constraints. With randomness affecting communication in either way, the communication environment can be represented as a Markov chain with stability understood as ergodicity. Stability of randomized communication algorithms can be considered in the queue-free model, in which a packet gets associated with a new station at the time of injection, and the station dies after the packet has been heard on the channel. Full sensing protocols were shown to fare well in this model; some protocols stable for injection rate slightly below  $1/2$  were developed, see Chlebus [21]. The model of a fixed set of stations with private queues was considered to be less radical, as queues appear to have a stabilizing effect. Håstad et al. [31], Al-Ammal et al. [2] and Goldberg et al. [29] studied bounds on the rates for which the binary exponential backoff was stable, as functions of the number of stations. For recent work related to the exponential backoff see the papers by Bender et al [16] and Bender et al [17], who proposed modifications to exponential backoff with the goal to improve some of its characteristics. Raghavan and Upfal [34] and Goldberg et al [30] proposed randomized broadcast algorithms based on different paradigms than those used in backoff protocols. Paper [21] includes a survey of randomized communication in multiple-access channels.

The methodology of adversarial queuing allows to capture the notion of stability of communica-

tion protocols without resorting to randomness and serves as a framework for worst-case bounds on performance of deterministic protocols. Borodin et al. [19] proposed this approach in the context of routing protocols in store-and-forward networks. This was followed by Andrews et al. [9], who emphasized the notion of universality in adversarial settings.

The adversarial approach to modeling communication proved to be inspirational and versatile. Álvarez et al. [3] applied adversarial models to capture phenomena related to routing of packets with varying priorities and failures in networks. Álvarez et al. [4] addressed the impact of link failures on stability of communication algorithms by way of modeling them in adversarial terms. Andrews and Zhang [11] considered adversarial networks in which nodes operate as switches connecting inputs with outputs, so that routed packets encounter additional congestion constraints at nodes when they compete with other packets for input and output ports and need to be queued when delayed. Andrews and Zhang [12] investigated routing and scheduling in adversarial wireless networks in which every node can transmit data to at most one neighboring node per time step and where data arrivals and transmission rates are governed by an adversary.

Worst-case packet latency of routing protocols for store-and-forward wired networks has been studied in the framework of adversarial queuing. Aiello et al. [1] demonstrated that polynomial packet latency can be achieved by a distributed algorithm even when the adversaries do not disclose the paths they assigned to packets in order to comply with congestion restrictions. Andrews et al. [10] studied packet latency of adversarial routing when the entire path of a packet is known at the source. Broder et al. [20] discussed conditions under which protocols effective for static routing provide bounded packet latency when applied in dynamic routing. Scheideler and Vöcking [37] investigated how to transform static store-and-forward routing algorithms, designed to handle packets injected at the same time, into efficient algorithms able to handle packets injected continuously into the network, so that packet delays in the static case are close to those occurring in the dynamic case. Rosén and Tsirkin [36] studied bounded packet delays against the ultimately powerful adversaries of rate 1.

Jamming in multiple-access channels and wireless networks is usually understood as disruptions occurring in individual rounds that prevent successful transmissions in spite of lack of collisions caused by concurrent interfering transmissions. Awerbuch et al. [14] studied jamming in multiple access channels in an adversarial setting with the goal to estimate saturation throughput of randomized protocols. Gilbert et al. [28] studied jammed transmissions on multiple access channel with the goal to optimize energy consumption per each transmitting station. Broadcasting on multi channels with jamming controlled by adversaries was studied by Chlebus et al. [22], Gilbert et al. [26], Gilbert et al. [27], and Meier et al. [33]. Richa et al. [35] considered broadcasting on wireless networks modeled as unit disc graphs with one communication channel, in which a constant fraction of rounds can be jammed.

## 2 Preliminaries

In this section, we review the technical specifications of the underlying model of communication and the algorithms. This is a preparation to consider packet latency of deterministic distributed communication algorithms, depending on a number of stations and an adversary that has the combined injection and jamming rates less than 1. In subsequent sections, we discuss specific deterministic communication algorithms, for which we estimate upper bounds on the queue size

and packet latency, as functions of the size of the system and an adversary at hand.

A communication environment we will study consists of some  $n$  stations attached to a channel. The stations receive packets continuously and their goal is to have them broadcast successfully. Each station is equipped with a private buffer space, which is organized as a queue. A station may hold multiple packets at a time, which are stored in this queue. The queues' capacities are assumed to be unbounded, in that a queue can accommodate an arbitrary number of packets. A *message* transmitted by a station consists of either a packet along with control bits or only control bits. We use the slotted model, in which time is partitioned into *rounds*. The size of a message and the duration of a round are calibrated such that a transmission of a message takes one round.

**Multiple access channel.** A message successfully transmitted on the communication medium is delivered to each station; we say that the message is *heard* by each station. A round when no message is heard on the channel is called *void*. A round may either be *jammed* or not; a round is *clear* when it is not jammed.

A broadcast system is said to be a *multiple-access channel without jamming* when the channel is never jammed and a message transmitted by a station is heard, instantaneously and by all the stations, if and only if the transmission does not overlap in time with any other transmissions. In the slotted model that we assume, a message is heard by all the stations in a round of transmission when exactly one station transmits in this round.

A broadcast system is said to be a *multiple-access channel with jamming* if a message transmitted by a station is heard, instantaneously and by all the stations, if and only if the transmission does not overlap in time with any other transmissions and the channel is not jammed during the transmission. In the slotted model of channels that we assume, a message is heard by all the stations in a round of transmission when exactly one station transmits in this round and the round is clear.

In each round, all the stations receives the same feedback from the channel. When a message is heard on the channel, then the message itself is such feedback. A round with no transmissions is said to be *silent*; in such a round, all the stations receive from the channel the feedback we call *silence*. Multiple transmissions in the same round result in conflict for access to the channel, which is called a *collision*. When a round is jammed then all the stations receive in this round the same feedback from the channel as in a round of collision.

Now we recapitulate the categorizations of rounds. When a round is void, that is, when no message is heard, then this is because of the following three possibilities. One possibility is that the round is silent, which means there is no transmission. The other possibility is that the round is jammed, then it does not matter whether there is any transmission or not. Finally, there may be a collision caused by multiple simultaneous transmissions. Stations cannot distinguish between a collision, caused by multiple simultaneous transmissions, and the channel being jammed in the round, in the sense that the channel is sensed in exactly the same manner in both cases.

We say that *collision detection* is available when stations can distinguish between a silence and either a collision or jamming by the feedback they receive from the channel; otherwise the channel is *without collision detection*. Next we consider in detail the four possible cases of a channel being either with jamming or not, and independently, being either with collision detection or not.

A channel is without jamming and without collision detection: a void round is perceived as silence, even when caused by a collision.

A channel is without jamming and with collision detection: a void round is perceived either as silence, when there is no transmission, or differently as a collision, when there are multiple simultaneous transmission.

A channel is with jamming and without collision detection: a void round is perceived as silence, and the stations cannot distinguish which of the following possible causes makes the round void, that is, either no message is transmitted or there is a collision or the round is jammed.

A channel is with jamming and with collision detection: a void round is either perceived as silence, which means there is no transmission, or as a collision, which means that either there are multiple simultaneous transmissions or the round is jammed.

**Adversarial model of packet injection.** We use the general leaky-bucket adversarial model, as considered in [9, 23]. An adversary is determined by injection rate and burstiness. Let a real number  $\rho$  satisfy  $0 < \rho \leq 1$ , and  $b$  be a non-negative integer; the *leaky-bucket adversary of type*  $(\rho, b)$  may inject at most  $\rho t + b$  packets into any set of stations in every contiguous segment of  $t > 0$  rounds. An adversary is said to be of *injection rate*  $\rho$  when it is of type  $(\rho, b)$ , for some  $b$ . *Burstiness* means the maximum number of packets that can be injected in one round. The adversary of type  $(\rho, b)$  has burstiness  $\lfloor b + 1 \rfloor$ .

**Knowledge.** A property of a system is said to be *known* when it can be referred to explicitly in a code of a communication algorithm. We assume throughout that the number of stations  $n$  is known to the stations. Each station has a unique integer name in  $[0, n - 1]$ , which it knows. When a station needs to be distinguished in a communication algorithm, for example to be the first one to transmit in an execution, then it is always the station with name 0.

**Definition of deterministic distributed broadcast algorithms.** Broadcast algorithms control timings of transmissions by individual stations. All the stations start simultaneously at the same round. The local queues at stations are under the FIFO discipline, which minimizes packet latency. A packet is never dropped by a station, unless it was heard on the channel.

A *state* of a station is determined by the private values of variables occurring in the code of the algorithm and the number of outstanding packets in its queue that still need to be transmitted. A *state transition* is a change of state in one round. An *execution* of an algorithm is a sequence of events occurring at consecutive rounds. An *event* in a round comprises the following actions at any station in the given order:

- a) a station either transmits a message or pauses, accordingly to its state,
- b) a station receives a feedback from the channel, in the form of either hearing a message or collision signal or silence, then
- c) new packets are injected into a station, if any, and finally
- d) a state transition occurs in a station.

A state transition depends on the state at the end of the previous round, the feedback from the channel in this round, and the packets injected in this round; it occurs as follows. If packets have been injected in this round then they are enqueued into the local queue. If the station has just

transmitted successfully in this round, then the transmitted packet is discarded and a new pending packet is obtained by dequeuing the local queue, unless it is empty. Finally, a message for the next round is prepared, if any will be attempted to be transmitted.

We categorize broadcast algorithms according to the terminology used in [23, 24]. All the algorithms considered in this paper are full sensing, in that nontrivial state transitions can occur at a station in any round, even when the station does not have pending packets to transmit. Algorithms that use control bits piggybacked on packets or send messages comprised of only control bits are called *adaptive*. When we refer to an algorithm simply as full sensing then this means that it is not adaptive.

A channel with jamming does not produce any special “interference” signal indicating that the round is jammed. It follows that a communication algorithm for channels without jamming can be executed, without any changes in its code, for channels with jamming.

**Jamming adversaries.** For channels with jamming, we consider adversaries that control both packet injections and jamming. Given real numbers  $\rho$  and  $\lambda$  in  $[0, 1]$  and integer  $b > 0$ , the *leaky-bucket jamming adversary of type  $(\rho, \lambda, b)$*  can inject at most  $\rho|\tau| + b$  packets and, independently, it can jam at most  $\lambda|\tau| + b$  rounds, in any contiguous segment  $\tau$  of  $|\tau| > 0$  rounds. For this adversary, we refer to  $\rho$  as the *injection rate* and to  $\lambda$  as the *jamming rate*. We can observe that the non-jamming adversary of type  $(\rho, b)$  is formally the same as the jamming adversary of type  $(\rho, 0, b)$ .

If  $\lambda = 1$  then every round could be jammed, making the channel dysfunctional. Therefore we assume that the jamming rate  $\lambda$  satisfies  $\lambda < 1$ . Stability is not achievable by a jamming adversary with injection rate  $\rho$  and the jamming rate  $\lambda$  satisfying  $\rho + \lambda > 1$ . To see this, observe that it is equivalent to  $\rho > 1 - \lambda$ , so when the adversary is jamming with the maximum capacity, then the bandwidth remaining for transmissions is  $1 - \lambda$ , while the injection rate is greater than  $1 - \lambda$ . It is possible to achieve stability in the case  $\rho + \lambda = 1$ , by adapting the approach for  $\rho = 1$  (and  $\lambda = 0$ ) in [23], but packet latency is then inherently unbounded. We assume throughout that  $\rho + \lambda < 1$ .

The number of packets that a jamming adversary can inject in one round is called its *injection burstiness*. This parameter equals  $\lfloor \rho + b \rfloor$  for a leaky-bucket adversary. The maximum continuous number of rounds that an adversary can jam is called its *jamming burstiness*. The leaky-bucket jamming adversary of type  $(\rho, \lambda, b)$  can jam at most  $\lfloor b/(1 - \lambda) \rfloor$  consecutive rounds, as the inequality  $\lambda x + b \geq x$  needs to hold for any such a number  $x$  of rounds.

The type of an adversary is not assumed to be known. The only exception to this rule in this paper occurs for a full-sensing algorithm that has an upper bound  $J$  on the jamming burstiness of an adversary as part of its code; this algorithm attains the claimed packet latency when the adversary’s jamming burstiness is at most  $J$ .

**Performance of broadcast algorithms.** The basic quality for a communication algorithm in a given adversarial environment is *stability*, understood in the sense that the number of packets in the queues at stations stays bounded at all times. For a stable algorithm in a communication environment, an upper bound on the number of packets waiting in queues is a natural performance metric, see [23, 24]. An algorithm is *universal* when it is stable for any injection rate smaller than 1. A sharper performance metric is that of *packet latency*; it denotes an upper bound on the time spent by a packet waiting in a queue, counting from the round of injection through the round when the packet is heard on the channel. For each algorithm that we consider in this paper, we give upper bounds for packet latency as functions of the number of stations  $n$  and the type  $(\rho, \lambda, b)$  of



a leaky-bucket adversary.

### 3 Specific Broadcast Algorithms

In this section we summarize the specifications of deterministic broadcast algorithms whose packet latency is analyzed in detail later. Some of these algorithms have already been considered in the literature and some are new.

**Three broadcast algorithms.** We start with three deterministic distributed algorithms for channels without jamming that are already known in the literature. These algorithms can be described as follows.

Algorithm ROUND-ROBIN-WITHHOLDING (RRW) is a full-sensing (non-adaptive) algorithm for channels without collision detection. It operates in a round-robin fashion, in that the stations gain access to the channel in the cyclic order of their names. Once a station gets access to the channel by transmitting successfully, it withholds the channel to unload all the packets in its queue. A silent round is a signal to the next station, in the cyclic order of names, to take over. Algorithm RRW was introduced in [24] and showed to be universal, that is, stable for injection rates smaller than 1.

Algorithm SEARCH-ROUND-ROBIN (SRR) is a full-sensing (non-adaptive) algorithm for channels with collision detection. Its execution proceeds as a systematic sweep across all the stations with the goal to identify these with packets, with such search performed in the cyclic order. When a station with a packet is identified, the station unloads all its packets one by one. A silent round triggers the sweep to be resumed. We apply binary search to identify the next station. The binary search is implemented using collision detection. When we inquire about a segment of stations, then all the stations with packets that are in the segment transmit in the round. A search is completed by a packet heard. A silence indicates that the segment is empty of stations with packets. A collision indicates that multiple stations are in the segment: this results in having the segment partitioned into two halves, with one segment processed next immediately while the other one is pushed on a stack to wait. A transition to the next segment occurs when the stack gets empty. Algorithm RRW was introduced in [24] and showed to be universal.

Algorithm MOVE-BIG-TO-FRONT (MBTF) is an adaptive algorithm for channels without collision detection. Each station maintains a list of all stations in its private memory. A list is initialized to be sorted in the increasing order of names of stations. The lists are manipulated in the same way by all the stations so their order is the same. The algorithm schedules exactly one station to transmit in a round, so collisions never occur. This is implemented by having a conceptual token assigned to stations, which is initially assigned to the first station on the list. A station with the token broadcasts a packet, if it has any, otherwise the round is silent. A station considers itself *big* in a round when it has at least  $n$  packets; such a station attaches a control bit to all packets it transmits to indicate this status. A big station is moved to the front of the list and it keeps the token for the next round. When a station that is not big transmits, or when it pauses due to a lack of packets while holding the token, the token is passed to the next station in the list ordered in a cyclic fashion. Algorithm MBTF was introduced in [23] and showed to be stable for injection rate 1.

**The “older-go-first” paradigm.** We obtain new algorithms by modifying RRW and SRR so that packets are categorized into “old” and “new.” Packets categorized as “new” become eligible for transmissions only after the packets categorized as “old” have been heard. Formally, an execution is structured as a sequence of conceptual phases, which are contiguous segments of rounds of dynamic length, and then the notions of old versus new packets are implemented through them.

A *phase* is defined as a full cycle made by the conceptual token to visit all stations. No additional communication is needed to mark a transition to a new phase. The “older-go-first” principle is implemented through phases by having packets injected in a given phase transmitted in the next phase. Similarly, the definition of old versus new packets is implemented by phases. For a given phase, packets are *old* when they have been injected in the previous phase, and the packets injected in the current phase are considered *new* for the duration of the phase. When a new phase begins, old packets have already been heard on the channel and new ones immediately graduate to old.

Algorithm OLDER-FIRST-ROUND-ROBIN-WITHHOLDING (OF-RRW) operates similarly as RRW, except that when a station gets access to the channel by transmitting successfully, then the station unloads all the old packets while the new packets stay in the queue.

Algorithm OLDER-FIRST-SEARCH-ROUND-ROBIN (OF-SRR) operates similarly as SRR, except that the search is for old packets only. When a new phase begins, then all old packets have been processed already.

**Algorithms for channels with jamming.** We introduce a new algorithm JAMMING-ROUND-ROBIN-WITHHOLDING( $J$ ), abbreviated as JRRW( $J$ ), designed for channels with jamming. The design of the algorithm is similar to that of RRW, the difference is in how the token is transferred from a station to the next one, in the cyclic order among stations. Just one void round should not trigger a transfer of the token, as it is the case in RRW, because not hearing a message may be caused by jamming.

The algorithm has a parameter  $J$  interpreted as the upper bound on jamming burstiness of the adversary. This parameter is used to facilitate transfer of control from a station to the next one by way of forwarding the token. The token is moved after hearing silence for precisely  $J + 1$  contiguous rounds, counting from either hearing a packet or moving the token; the former indicates that the transmitting station exhausted its queue while the latter indicates that the queue is empty. More precisely, every station maintains a private counter of void rounds. The counters show the same value across the system, as they are updated in exactly the same way determined only by the feedback from the channel. A void round results in incrementing the counter by 1. The token is moved to the next station when the counter reaches  $J + 1$ . When either a packet is heard or the token is moved then the counter is zeroed.

Algorithm OLDER-FIRST-JAMMING-ROUND-ROBIN-WITHHOLDING( $J$ ), abbreviated OF-JRRW( $J$ ), is obtained from JRRW( $J$ ) similarly as OF-RRW is obtained from RRW. An execution is structured into phases, and packets are categorized into old and new, with the same rule to graduate packets from new to old. When a token visits a station, then only the old packets are transmitted while the new ones will be transmitted during the next visit by the token.

**Structural properties of algorithms.** We say that a communication algorithm designed for a channel without jamming is a *token algorithm* if it uses a virtual token to avoid collisions. Such a token is always held by some station and only the station that holds the token can transmit.

Algorithms RRW, OF-RRW, JRRW, OF-JRRW, and MBTF are token ones. We can take any token algorithm for channels without jamming and adapt it to the model with jamming in the following manner. If there is a packet to be transmitted by a station in the original algorithm, then the modified algorithm has a packet transmitted as well, otherwise just a control bit is transmitted. A round in which only a control bit is transmitted by a modified token algorithm is called *control round* otherwise it is a *packet round*. The effect of sending control bits in control rounds is that if a round is not jammed then a message is heard in this round; this message is either just a control bit or it includes a packet. This approach creates virtual collisions in jammed rounds, so when a void round occurs then this round is jammed as otherwise a message would have been heard. Once a communication algorithm can identify jammed rounds, we may ignore their impact on the flow of control. The resulting algorithm is adaptive. We will consider such modified version of the full-sensing algorithms RRW and OF-RRW, denoting them by C-RRW and OFC-RRW, respectively. Note that algorithm MBTF works by having a station with the token send a message even if the station does not have a packet, so enforcing additional control rounds is not needed for this algorithm to convert it into one creating virtual collisions.

Algorithms with executions structured into phases are referred to as *phase algorithms*. These include RRW, OF-RRW, JRRW, OF-JRRW, C-RRW, and OFC-RRW. When the older-go-first paradigm is used in such a communication algorithm then it is called the *older-go-first version* of the algorithm, otherwise it is the *regular version* of the algorithm. In particular, RRW, JRRW and C-RRW are regular phase algorithms, while OF-RRW, OF-JRRW and OFC-RRW are older-go-first phase algorithms.

## 4 Full Sensing Algorithms with Jamming

We show that a bounded worst-case packet latency is achievable by full sensing algorithms against adversaries for whom jamming burstiness is at most a given bound  $J$ , while  $J$  is a part of code. On the other hand, we will demonstrate later that adaptive algorithms can achieve bounded packet latency without restricting jamming burstiness of adversaries in their code. The full sensing algorithms we consider are OF-JRRW( $J$ ) and JRRW( $J$ ). Algorithms OF-JRRW( $J$ ) and JRRW( $J$ ) include the parameter  $J$  as a part of code, but the value of  $J$  does not occur in the upper bounds on packet latency in Theorems 1 and 2.

**Lemma 1** *Consider an execution of algorithm OF-JRRW( $J$ ) against a leaky-bucket adversary of jamming rate  $\lambda$ , burstiness  $b$ , and jamming burstiness at most  $J$ . If there are  $x$  old packets in the queues in a round, then at least  $x$  packets are transmitted within the next  $(x + n(J + 1) + b)/(1 - \lambda)$  rounds.*

**Proof:** It takes  $n$  intervals of  $J + 1$  void rounds each for the token to make a full cycle and so to visit every station with old packets. It is advantageous for the adversary not to jam the channel during these rounds. Therefore, at most  $n(J + 1) + x$  clear rounds are needed to hear the  $x$  packets. Consider a contiguous time segment of  $z$  rounds in which some  $x$  packets are heard. At most  $z\lambda + b$  of these  $z$  rounds can be jammed. Therefore, the following inequality holds:

$$z \leq n(J + 1) + x + z\lambda + b .$$

Solving for  $z$ , we obtain

$$z \leq \frac{x + n(J+1) + b}{1 - \lambda}$$

as the bound on the length of a contiguous time interval in which at least  $x$  packets are heard.  $\square$

**Theorem 1** *The packet latency of algorithm OF-JRRW( $J$ ) is  $\mathcal{O}(\frac{bn}{(1-\lambda)(1-\rho-\lambda)})$ , when executed against a jamming adversary of type  $(\rho, \lambda, b)$  such that its jamming burstiness is at most  $J$ .*

**Proof:** Let  $t_i$  be the duration of phase  $i$  and  $q_i$  be the number of old packets in the beginning of phase  $i$ , for  $i \geq 1$ . The following two estimates lead to a recurrence for the numbers  $t_i$ . One is

$$q_{i+1} \leq \rho t_i + b, \quad (1)$$

which follows from the definitions of old packets and of type  $(\rho, \lambda, b)$  of the adversary. The other estimate is

$$t_{i+1} \leq \frac{n(J+1) + q_{i+1} + b}{1 - \lambda}, \quad (2)$$

which follows from Lemma 1. Let us denote  $n(J+1) = a$ . Substitute (1) into (2) to obtain

$$\begin{aligned} t_{i+1} &\leq \frac{a + q_{i+1} + b}{1 - \lambda} \\ &\leq \frac{a}{1 - \lambda} + \frac{b}{1 - \lambda} + \frac{\rho t_i + b}{1 - \lambda} \\ &= \frac{a}{1 - \lambda} + \frac{2b}{1 - \lambda} + \frac{\rho}{1 - \lambda} t_i \\ &\leq c + d t_i, \end{aligned}$$

for  $c = \frac{a+2b}{1-\lambda}$  and  $d = \frac{\rho}{1-\lambda}$ . Note that  $d < 1$ , as  $\rho < 1 - \lambda$ .

We will find an upper bound on the duration of a phase by iterating the recurrence  $t_{i+1} \leq c + d t_i$ . To this end, it is sufficient to inspect the following sequence of consecutive bounds on the lengths of the initial phases

$$t_1 \leq c, \quad t_2 \leq c + d c, \quad t_3 \leq c + d c + d^2 c, \quad \dots$$

to discover a general pattern to the effect

$$t_{i+1} \leq c + d c + d^2 c + \dots d^i c \leq \frac{c}{1 - d}. \quad (3)$$

After substituting  $c = \frac{a+2b}{1-\lambda}$  and  $d = \frac{\rho}{1-\lambda}$  into (3), we obtain

$$\begin{aligned} t_i &\leq \frac{a + 2b}{1 - \lambda} \cdot \frac{1}{1 - \frac{\rho}{1 - \lambda}} \\ &\leq \frac{a + 2b}{1 - \lambda} \cdot \frac{1 - \lambda}{1 - \rho - \lambda} \\ &\leq \frac{a + 2b}{1 - \rho - \lambda}. \end{aligned} \quad (4)$$

Now, replace  $a$  by  $n(J+1)$  in (4) to expand it into the following inequality:

$$t_i \leq \frac{n(J+1) + 2b}{1 - \rho - \lambda}. \quad (5)$$

Apply the estimate  $J \leq b/(1 - \lambda)$  to (5) to obtain that  $t_i$  is at most

$$\begin{aligned} t_i &\leq \frac{n(\frac{b}{1-\lambda} + 1) + 2b}{1 - \rho - \lambda} \\ &\leq \frac{2(bn + (n + b)(1 - \lambda))}{(1 - \lambda)(1 - \rho - \lambda)}, \end{aligned} \quad (6)$$

which is a bound on the duration of a phase that depends only on the type of the adversary, without involving  $J$ . The bound on packet latency we seek is twice that in (6), as a packet stays queued for at most two consecutive phases.  $\square$

Next, we give an upper bound on the packet latency of algorithm JRRW( $J$ ). The proof of Theorem 2 is obtained by comparing packet latency of algorithm JRRW( $J$ ) with that of algorithm OF-JRRW( $J$ ) as given in Theorem 1.

**Theorem 2** *The packet latency of algorithm JRRW( $J$ ) is  $\mathcal{O}(\frac{bn}{(1-\lambda)(1-\rho-\lambda)^2})$ , when executed against a jamming adversary of type  $(\rho, \lambda, b)$  such that its jamming burstiness is at most  $J$ .*

**Proof:** We will relate executions of algorithms JRRW( $J$ ) and OF-JRRW( $J$ ) determined by some injection and jamming pattern of the adversary. Let  $s_i$  and  $t_i$  be bounds on the length of phase  $i$  of algorithms OF-JRRW( $J$ ) and JRRW( $J$ ), respectively, when run against the considered adversarial pattern of injections and jamming.

A phase  $i$  of algorithm OF-JRRW( $J$ ) takes  $s_i$  rounds. When algorithm JRRW( $J$ ) is executed, the total number of rounds  $t_i$  in phase  $i$  is at most

$$s_i + s_i(\rho + \lambda) + s_i(\rho + \lambda)^2 + \dots = \frac{s_i}{1 - (\rho + \lambda)}. \quad (7)$$

We obtain that the phase's length of algorithm JRRW( $J$ ) differs from that of algorithm OF-JRRW( $J$ ) by at most a factor of  $\frac{1}{1-\rho-\lambda}$ .

Algorithms JRRW( $J$ ) and OF-JRRW( $J$ ) share the property that a packet is transmitted in at most two consecutive phases, where the first of them is determined by the injection of the packet. The bound on packet latency given in Theorem 1 is of twice the length of a phase of algorithm OF-JRRW( $J$ ). Similarly, a bound on twice the length of a phase of algorithm JRRW( $J$ ) is a bound on packet latency. It follows that a bound on packet latency of algorithm JRRW( $J$ ) can be obtained by multiplying the bound given in Theorem 1 by  $\frac{1}{1-\rho-\lambda}$ .  $\square$

The upper bounds on packet latency given in Theorems 2 and 1 differ by the multiplicity of the factor  $1 - \rho - \lambda$  occurring in the denominator. This difference between the two bounds reflects the benefit of the paradigm “older-go-first” applied in the design of algorithm OF-JRRW( $J$ ), as compared to algorithm JRRW( $J$ ).

We want to show next that the bound of Theorem 2 is tight. To this end, consider algorithm JRRW( $J$ ) in a channel with jamming. The adversary will inject packets and jam the channel at full power, subject to constraints imposed by its type. The number of clear void rounds in each phase is  $a = n(J + 1)$ . The number of packets injected due to these void rounds is  $a\rho$ , and it takes  $a(\rho + \lambda)$  rounds to transmit them. In the first phase,  $a\rho$  packets are transmitted in  $a(\rho + \lambda)$  rounds and a queue of  $a(\rho + \lambda)\rho$  packets builds up in station  $n - 1$ . In the second phase, there are  $a\rho$

and  $n(\rho + \lambda)\rho$  packets transmitted by stations  $n$  and  $n - 1$ , respectively; this takes time  $a(\rho + \lambda)^2$  and results in  $a(\rho + \lambda)^2\rho$  packets queued at station  $n - 2$ . This process continues until phase  $n - 2$  in which the adversary's behavior changes. The difference is that the adversary injects just one packet into station 2 after the token has passed through that station. What follows is phase  $n - 1$  in which the queue at station 1 is  $\Omega(\frac{a+b}{1-\rho-\lambda})$ . Let the adversary inject packets at full power only into station 1 in this phase. Consider the latency of packet in station 2. The packets in station 1 are unloaded by withholding the channel while the adversary keeps injecting only into the transmitting station. Let us assign a suitably large  $J$  to the adversary, for example,  $J \geq b/(2(1 - \lambda))$  will do. The latency of the only packet in station 2 is estimated as being

$$\begin{aligned} \Omega\left(\frac{n(J+1)+b}{(1-\rho-\lambda)^2}\right) &= \Omega\left(\frac{n(\frac{b}{2(1-\lambda)}+1)+b}{(1-\rho-\lambda)^2}\right) \\ &= \Omega\left(\frac{bn}{(1-\lambda)(1-\rho-\lambda)^2}\right). \end{aligned}$$

This shows that the upper bound given in Theorem 2 is tight.

Regarding the bound of Theorem 1, one possible way to argue about its tightness is to examine its proof to see that the derivation can be mimicked by adversary's actions. Another approach to argue that the bound in Theorem 1 is tight is to observe that the bound in Theorem 2 was obtained from the bound in Theorem 1 by multiplying it by the factor  $\frac{1}{1-\rho-\lambda}$ , so that any improvement in the bound in Theorem 1 would be reflected in an improvement in the bound in Theorem 2, which has been shown to be tight.

**Knowledge of jamming burstiness.** We have shown that a full sensing algorithm achieves bounded packet latency for  $\rho + \lambda < 1$  when an upper bound on jamming burstiness is a part of code. Next, we hypothesize that this is unavoidable and reflects the utmost power of full sensing (non-adaptive) algorithms.

**Conjecture 1** *No full sensing but non-adaptive algorithm can be stable against all jamming adversaries with the injection rate  $\rho$  and jamming rate  $\lambda$  satisfying  $\rho + \lambda < 1$ .*

## 5 Adaptive Algorithms with Jamming

We give upper bounds on packet latency for the following three adaptive algorithms: C-RRW, OFC-RRW, and MBTF. The bounds are similar to those obtained for their full sensing counterparts in Theorems 1 and 2. The apparent relative strength of adaptive algorithms is reflected in their bounds shedding the factor  $1 - \lambda$  in the denominators. Each of these adaptive algorithms is stable for any jamming burstiness, unlike the full sensing algorithms we considered, which have in their codes a bound on jamming burstiness they can handle in a stable manner.

**Lemma 2** *Consider an execution of algorithm OFC-RRW against a leaky-bucket adversary of some type  $(\rho, \lambda, b)$ . If there are  $x$  old packets in the system in some round, then at least  $x$  packets are transmitted within the next  $\frac{x+n+b}{1-\lambda}$  rounds.*

**Proof:** It takes  $n$  control rounds for the token to pass through all  $n$  stations. Consider a contiguous time segment of  $y$  rounds in which some  $x$  packets are heard. At most  $y\lambda + b$  of these  $y$  rounds can

be jammed. It follows that the following inequality holds:

$$y \leq n + x + y\lambda + b .$$

Solving for  $y$  yields the upper bound

$$y \leq \frac{x + n + b}{1 - \lambda} ,$$

as the bound on the length of a contiguous time interval in which at least  $x$  packets are heard.  $\square$

**Theorem 3** *The packet latency of algorithm OFC-RRW is  $\mathcal{O}(\frac{n+b}{1-\rho-\lambda})$  when executed against the jamming adversary of type  $(\rho, \lambda, b)$ .*

**Proof:** Let  $t_i$  denote the duration of phase  $i$  and  $q_i$  be the number of old packets in the beginning of phase  $i$ , for  $i \geq 1$ . We use the following two estimates to derive a recurrence for the numbers  $t_i$ . One is

$$q_{i+1} \leq \rho t_i + b , \tag{8}$$

which follows from the definition of old packets and the adversary of type  $(\rho, \lambda, b)$ . The other is

$$t_{i+1} \leq \frac{q_{i+1} + n + b}{1 - \lambda} , \tag{9}$$

which follows from Lemma 2. Using the abbreviations  $c = \frac{n+2b}{1-\lambda}$  and  $d = \frac{\rho}{1-\lambda}$ , we substitute (8) into (9) to obtain

$$\begin{aligned} t_{i+1} &\leq \frac{n+b}{1-\lambda} + \frac{\rho t_i + b}{1-\lambda} \\ &= \frac{n+2b}{1-\lambda} + \frac{\rho}{1-\lambda} t_i \\ &\leq c + d t_i . \end{aligned}$$

To find an upper bound on the duration of a phase, we iterate the recurrence  $t_{i+1} \leq c + d t_i$ , which produces

$$t_{i+1} \leq c + d c + d^2 c + \dots + d^i c \leq \frac{c}{1-d} . \tag{10}$$

After substituting  $c = \frac{n+2b}{1-\lambda}$  and  $d = \frac{\rho}{1-\lambda}$  into (3), we obtain, by algebraic manipulations, the following estimate:

$$\begin{aligned} t_i &\leq \frac{n+2b}{1-\lambda} \cdot \frac{1}{1 - \frac{\rho}{1-\lambda}} \\ &\leq \frac{n+2b}{1-\lambda} \cdot \frac{1-\lambda}{1-\rho-\lambda} \\ &\leq \frac{2(n+b)}{1-\rho-\lambda} . \end{aligned} \tag{11}$$

The bound on packet latency we seek is twice that in (11), because a packet is queued for at most two consecutive phases.  $\square$

**Theorem 4** *The packet latency of algorithm C-RRW is  $\mathcal{O}(\frac{n+b}{(1-\rho-\lambda)^2})$ , when executed against the jamming adversary of type  $(\rho, \lambda, b)$ .*

**Proof:** We compare packet latency of algorithm C-RRW to that of algorithm OFC-RRW. Consider executions of algorithms C-RRW and OFC-RRW for the same injection and jamming pattern of the adversary. Let  $s_i$  and  $t_i$  be the bounds on the length of phase  $i$  of algorithms C-RRW and OFC-RRW, respectively.

Phase  $i$  of C-RRW takes  $s_i$  rounds. When algorithm OFC-RRW is executed, then its phase  $i$  may take up to the following number of rounds:

$$s_i + s_i(\rho + \lambda) + s_i(\rho + \lambda)^2 + \dots = \frac{s_i}{1 - (\rho + \lambda)}.$$

So the phase's length of algorithm C-RRW differs from that of algorithm OFC-RRW by a factor of at most  $\frac{1}{1-\rho-\lambda}$ . An injected packet is transmitted in at most two phases of an execution, for each of the two considered algorithms. The bound of Theorem 3 is twice the bound on the duration of a phase. It follows that a bound on packet latency of algorithm C-RRW can be obtained by multiplying the bound given in Theorem 3 by  $\frac{1}{1-\rho-\lambda}$ .  $\square$

The tightness of the bound on packet latency given in Theorem 4 can be established similarly as that for Theorem 2. To this end, replace  $n(J+1)$  by  $n$  in the tightness argument for the bound of Theorem 2 given in Section 4 to obtain  $\Omega(\frac{n+b}{(1-\rho-\lambda)^2})$  as a bound. The tightness of the upper bound of Theorem 3 follows from the property that the derivation of the upper bound can be closely mimicked by the adversary. Alternatively, we observe that an improvement of the bound in Theorem 3 would lead to an improvement to the bound in Theorem 4, which has already been shown to be tight.

**Algorithm Move-Big-To-Front.** Next we estimate the packet latency of algorithm MBTF against jamming. The analysis we give resorts to estimates of the number of packets stored in the queues at all times.

Let a traversal of the token, starting at the front end of the list and ending again at the front station of the list, be called a *pass* of the token. A pass is concluded by either discovering a new big station or traversing the whole list.

We monitor the number of packets in the queues at the end of a pass, to see if the pass contributed to an increase of the number of packets stored in the queues or not. When the number of packets in queues at the end of a pass is bigger than at the end of the previous pass then such a pass is called *increasing*, otherwise it is *non-increasing*.

Additionally, we partition passes into two categories, depending on whether a big station is discovered in the pass or not. When a big station is discovered in a pass then such a pass is called *big* and otherwise it is called *small*. To make this terminology precise, we clarify how actions of a newly discovered big station are categorized, with respect to two consecutive passes. A discovery of a big station concludes a pass, but this big station just found does not transmit in this pass, because we consider the pass as already finished. The next pass begins by a transmission of the newly discovered big station, just after it has been moved up to the front position in the list.

**Lemma 3** *When algorithm MBTF is executed by  $n$  stations against the jamming adversary of type  $(\rho, \lambda, b)$  then the number of packets stored in queues in each round is at most  $\frac{2\rho n(n+b)}{(1-\lambda)(1-\rho-\lambda)} + \mathcal{O}(bn)$ .*

**Proof:** We first investigate how many packets can be accumulated in queues when small passes occur. It is sufficient to consider increasing small passes only, as otherwise the previous passes contribute bigger counts of packets.



A pass that is not jammed takes  $n$  rounds, but with jamming it may take longer, as the jammed rounds slow the algorithm down. The number of rounds in a small pass is at most

$$b + n + n\lambda + n\lambda^2 + n\lambda^3 \dots \leq \frac{n}{1-\lambda} + b .$$

Therefore at most  $(\frac{n}{1-\lambda} + b)\rho + b$  packets get injected during a small pass. This is also an upper bound on the number of stations with packets during an increasing small pass, because if there were more such stations than each of them would transmit a packet during a pass. Each such a station has at most  $n - 1$  packets, because the pass is small. It follows that a small pass is either increasing or the number of packets in the queues, when the pass is over, is at most

$$((\frac{n}{1-\lambda} + b)\rho + b)(n - 1) = \frac{\rho n(n + b)}{1 - \lambda} + \mathcal{O}(bn) . \quad (12)$$

Next we consider the question how much above the upper bound (12) can the queues grow during big passes. Consider a time interval  $T$  when a maximum number of packets is accrued while some  $k$  stations are discovered at least once as big. To obtain estimates from above on the total number of packets in the queues, we conservatively assume the following:

- (i) when a small station is passed by the token then there are no packets in the queue of the station and the resulting round is a control one, and
- (ii) when a big station is discovered and moved up to the front of the list, then only one packet is transmitted by the station and then the token immediately moves on to the next station.

Consider a series of consecutive big passes. The number of control rounds in any big pass is at most  $n - 1$  followed by a big station moved to the front of the list. In the first pass, there are at most  $n - 1$  control rounds. In the second pass, the station discovered big in the first pass transmits a packet, which is next followed by at most  $n - 1$  control rounds until a new big station is discovered. In the third pass, the two big stations discovered so far transmit at least one packet each, as they are at the front of the list, which is followed by at most  $n - 2$  control rounds before the third big station is discovered. This pattern continues until the last pass which begins by all the  $k$  big stations residing in the initial segment of the list and so each of them transmitting one packet in this pass followed by at most  $n - k$  control rounds. Let  $t$  be the last round when the above last control round among at most  $n - k$  occurs.

When the next pass begins in round  $t + 1$ , then either the pass is small or a big station among the first  $k$  stations in the list is discovered, as these are the stations discovered as big in time interval  $T$ . In the former case, the upper bound (12) on the number of packets in the queues applies. In the latter one, there are no control rounds during the pass. It follows that an upper bound is obtained by estimating from above an increase of packets in the time interval  $T$  by round  $t$  and next adding to it the bound (12) as a possible starting point of the process of increasing queues in  $T$ .

Next we estimate the increase of packets in  $T$  by round  $t$ . There are at most  $(k + 1)(n - 1)$  control rounds and  $1 + 2 + 3 + \dots + k = k(k + 1)/2$  packets transmitted in packet rounds. At the same time, the adversary may be injecting packets and jamming rounds, with a net increase contributed by control and jammed rounds. The increase is at most

$$\frac{\rho}{1-\lambda} \left[ (k + 1)(n - 1) + \frac{k(k + 1)}{2} \right] + b - \frac{k(k + 1)}{2} = \frac{\rho}{1-\lambda} kn - \frac{1}{2} \left( 1 - \frac{\rho}{1-\lambda} \right) k^2 + \mathcal{O}(n + b) .$$

We seek to maximize the function

$$f(k) = \frac{\rho}{1-\lambda}kn - \frac{1}{2}\left(1 - \frac{\rho}{1-\lambda}\right)k^2.$$

The maximum occurs at the argument

$$k_{\max} = \frac{\rho n}{1 - \rho - \lambda},$$

which can be justified by the standard maximum finding procedure based on differentiation. By algebraic manipulation, we obtain that the values  $f(k)$  are at most

$$f(k_{\max}) \leq \frac{\rho n^2}{2(1-\lambda)(1-\rho-\lambda)}$$

and the increase of packets in  $T$  by round  $t$  is at most this number plus  $\mathcal{O}(n+b)$ . We combine this estimate with bound (12) to obtain the estimate

$$\frac{\rho n(n+b)}{1-\lambda} + \frac{\rho n^2}{2(1-\lambda)(1-\rho-\lambda)} + \mathcal{O}(bn) \leq \frac{2\rho n(n+b)}{(1-\lambda)(1-\rho-\lambda)} + \mathcal{O}(bn)$$

as a bound on the total number of queued packets. These estimates are with respect to what are the numbers of packets at ends of passes. Possible fluctuations of the number of packets in queues during a pass cannot be more than  $n+b$ , which is within the magnitude of the asymptotic component  $\mathcal{O}(bn)$  of the bound.  $\square$

When big stations get discovered then the regular round-robin pattern of token traversal is disturbed. Suppose a station  $i$  is discovered as big, which results in moving the station to the front. The clear rounds that follow, before the token either moves to position  $i+1$  or a new big station is discovered, whichever occurs first, are called *delay rounds*. These rounds are spent, first, on transmissions by the new first station, to bring the number of packets in its queue down to  $n-1$ , and next, the token needs  $i$  additional clear rounds to move to the station at position  $i+1$ . Given a round  $t$  in which a packet  $p$  is injected, let us take a snapshot of the queues in this round. Let  $q_i$  be the number of packets in a station  $i$  in this snapshot, for  $1 \leq i \leq n$ . We associate *credit* of the amount  $\max\{0, q_i - (n-i)\}$  with station  $i$  at this round. A station  $i$  has a positive credit when the inequality  $q_i \geq n-i+1$  holds. In particular, a big station  $i$  has credit  $q_i + i - n \geq i$ . Let  $C(n, t)$  denote the sum of all the credits of the stations in round  $t$ . We consider credit only with respect to the packets already in the queues in round  $t$ , unless stated otherwise.

**Lemma 4** *If discovering a big station in round  $t$  delays the token by some  $x$  rounds, excluding jammed rounds, then the amounts of credit satisfy  $C(n, t+x) = C(n, t) - x$ .*

**Proof:** We refer to stations by their positions just before the shift, unless stated otherwise. When a big station  $i$  is moved up to the front of the list, its credit gets decreased first by  $i-1$  by the change of position in the list, and next by the amount equal to the number of transmitted packets by the new first station. More precisely,  $q_i - (n-1)$  packets are transmitted in these many rounds to decrease credit of the new first station from  $q_i + 1 - n$  to zero, a unit of credit spent in each round.

The decrease of the amount of credit by  $i-1$  is to pay for the travel of the token to position  $i+1$ . There is a problem of what happens to the credit at the traversed stations that changed their

positions. The stations at the original positions 1 through  $i - 1$  get shifted by one position down the list to occupy the positions 2 through  $i$ . Consider such a station  $j$ , for  $1 \leq j \leq i - 1$ , when it is shifted one position down the list. Its credit stays equal to 0, if  $q_j < n - j$ , but it gets incremented by 1 if  $q_j \geq n - j$ , as it then equals  $q_j + (j + 1) - n \geq 1$ .

When a packet is transmitted by a station that does not hold any credit, then this does not affect the total amount of credit, as the credit contributed by this station stays equal to zero. A packet transmitted by a station with a positive credit decrements the credit by 1, which restores the amount of credit held by the station before the shift down. This means that the total decrease of the amount of credit equals the number of rounds in the time period of delay.  $\square$

**Theorem 5** *The packet latency of algorithm MBTF is at most  $\frac{3n(n+b)}{(1-\lambda)(1-\rho-\lambda)} + \mathcal{O}\left(\frac{bn}{1-\lambda}\right)$  when executed against the jamming adversary of type  $(\rho, \lambda, b)$ .*

**Proof:** Let a packet  $p$  be injected in some round  $t$  into station  $i$ . Let  $S(n, t)$  be an upper bound on the number of rounds that packet  $p$  spends waiting in the queue at  $i$  to be heard when no big station is discovered by the time packet  $p$  is eventually transmitted. Some additional waiting time for packet  $p$  is contributed by discoveries of big stations and the resulting delays; let us denote by  $T(n, t)$  the number of rounds by which  $p$  is delayed this way. The total delay of  $p$  is at most  $S(n, t) + T(n, t)$ .

First, we find upper bounds on the expression  $S(n, t) + T(n, t)$  that does not depend on  $t$  but only on  $n$ . The inequality

$$S(n, t) \leq \frac{n(n-1)}{1-\lambda} \quad (13)$$

holds because there are at most  $n - 1$  packets in the queue of  $i$  and each pass of the token takes at most  $\frac{n}{1-\lambda}$  rounds. The following inequality

$$T(n, t) \leq \frac{C(n, t)}{1-\lambda} \quad (14)$$

holds because of Lemma 4 and the fact that iterated delays due to jamming contribute the factor  $1/(1-\lambda)$ .

Observe that  $C(n, t)$  is upper bounded by the number of packets in the queues in round  $t$ , by the definition of credit. Therefore, we obtain the following inequality by Lemma 3:

$$C(n, t) \leq \frac{2\rho n(n+b)}{(1-\lambda)(1-\rho-\lambda)} + \mathcal{O}(bn) .$$

This, along with the estimate (14), implies the following bound:

$$T(n, t) \leq \frac{2\rho n(n+b)}{(1-\lambda)^2(1-\rho-\lambda)} + \mathcal{O}\left(\frac{bn}{1-\lambda}\right) . \quad (15)$$

Combine (13) and (15) to obtain that a packet waits at most

$$\begin{aligned} & \frac{n(n-1)}{1-\lambda} + \frac{2\rho n(n+b)}{(1-\lambda)^2(1-\rho-\lambda)} + \mathcal{O}\left(\frac{bn}{1-\lambda}\right) \\ & \leq \frac{3n(n+b)}{(1-\lambda)(1-\rho-\lambda)} + \mathcal{O}\left(\frac{bn}{1-\lambda}\right) \end{aligned}$$

rounds, where we used the inequalities  $\rho < 1 - \lambda$  and  $1 - \rho - \lambda < 1$ .  $\square$

The tightness of the bound given in Theorem 5 can be established as follows. Let  $(\rho, \lambda, b)$  be the type of an adversary, where we assume  $0 < \rho < 1$  to be sufficiently close to 1 to simplify the argument. Consider first the case of no jamming, that is, when  $\lambda = 0$ . The adversary begins an execution by building a queue of  $n - 1$  packets at the last station in the list. Next the adversary will work to block the token reaching the last station by building big stations whose discovery makes the token go back to the front of the list. Let us call a station *critical* when it is the station into which the adversary is currently injecting packets to make it big. Once a critical station stores  $n$  packets, the adversary switches to the immediately preceding station to make it big by injecting packets into it; this station thereby becomes critical. Each new big station delays the token getting to the last station in the list by  $n - 1$  rounds. Next we calculate how many big stations can be built in a sequence without the token reaching the last station. Suppose the token is at the first station when the adversary begins working on a new critical stations. When the token reaches this station then it has about  $n\rho$  packets in its queue, with  $(1 - \rho)n$  packets missing to become big. During the next token's traversal the critical station becomes big and when the token reaches it the previous station misses  $2(1 - \rho)n$  packets to also be big. This patterns keeps repeating, for example, during the next token's traversal the critical station becomes big and when the token reaches it the previous station needs only  $3(1 - \rho)n$  packets to become big. This continues about  $\frac{n}{(1 - \rho)n} = \frac{1}{1 - \rho}$  times until the token manages to reach the last station in the list. The queue at the last station has  $n - 1$  packets, so the last of them will wait for the preceding  $n - 2$  packets to be heard. Each such a packet is delayed by about  $\frac{1}{1 - \rho}$  token traversals each taking  $n - 1$  rounds. The delay of some packet is therefore  $\Omega(\frac{n^2}{1 - \rho})$ .

Next, let us adapt this construction when the adversary has a jamming power. When the adversary applies the strategy to jam a round whenever possible, this slows down token traversal by a factor of  $\frac{1}{1 - \lambda}$ . The first critical station stores  $\frac{\rho n}{1 - \lambda}$  packets, rather than  $\rho n$  without jamming, when the token passes it for the first time; this makes it to have about

$$n(1 - \frac{\rho}{1 - \lambda}) = n \frac{1 - \lambda - \rho}{1 - \lambda}$$

packets short of becoming big. New big stations are discovered about

$$\frac{n}{1 - \lambda} \div \frac{n(1 - \lambda - \rho)}{1 - \lambda} = \frac{1}{1 - \lambda - \rho}$$

times before the token reaches the last station. We obtain that the delay of some packet is  $\Omega(\frac{n^2}{(1 - \lambda)(1 - \lambda - \rho)})$ .

## 6 Channels without Jamming

In this Section, we consider deterministic distributed algorithms for channels with no jamming. We study the performance of full sensing and adaptive algorithms against leaky-bucket adversaries with injection rates  $\rho < 1$ . For each of the algorithms we consider, we give upper bounds for packet latency as functions of the number of stations  $n$  and the type  $(\rho, b)$  of a leaky-bucket adversary.

**Full sensing algorithms without collision detection.** We begin with estimating packet latency for full sensing algorithms RRW and OF-RRW. They operate similarly as the adaptive algorithms C-RRW and OFC-RRW for a channel with jamming, respectively. The difference is

in how the virtual token moves from station to station and what happens in jammed rounds. In adaptive algorithms with jamming, the virtual token is moved when the station with token transmits a control bit. In non-adaptive algorithms, the token is moved when the station holding the token pauses. In adaptive algorithms with jamming, the jammed rounds is perceived as silence and just ignored. This facilitates translating the bounds for jammed channel to the channel without jamming, for the respective algorithms. The following results are obtained by such translations.

**Corollary 1** *The packet latency of algorithm OF-RRW is  $\mathcal{O}(\frac{n+b}{1-\rho})$  when executed against the adversary of type  $(\rho, b)$ .*

**Proof:** The upper bound on packet latency given in Theorem 3 becomes  $\mathcal{O}(\frac{n+b}{1-\rho})$  for  $\lambda = 0$ .  $\square$

**Corollary 2** *The packet latency of algorithm RRW is  $\mathcal{O}(\frac{n+b}{(1-\rho)^2})$  when executed against the adversary of type  $(\rho, b)$ .*

**Proof:** The upper bound on packet latency given in Theorem 4 becomes  $\mathcal{O}((n+b)/(1-\rho)^2)$  for  $\lambda = 0$ .  $\square$

These bounds are tight. The argument for tightness of bounds for algorithms for channels with jamming hold for the full sensing algorithms for channels without jamming when the jamming rate is  $\lambda = 0$ .

**Full sensing algorithms with collision detection.** We continue with estimates of packet latency of algorithms SEARCH-ROUND-ROBIN (SRR) and OLDER-FIRST-SEARCH-ROUND-ROBIN (OF-SRR), both of which use collision detection. Executions are partitioned into phases, similarly as when considering any older-go-first type of algorithm. In the case of algorithm SEARCH-ROUND-ROBIN, a phase denotes one full sweep of searches through the range of names of stations.

We begin with a technical estimate that will be used in proving bounds on packet latency. Let  $\lg x$  denote  $\lceil \log_2 x \rceil$ .

**Lemma 5** *If there are  $x \geq 1$  packets in the system in a round then, for each  $1 \leq y \leq x$ , algorithms SRR and OF-SRR make  $y$  packets heard in the next  $\min(y \lg n, 2n + y)$  rounds.*

**Proof:** We argue first that each of the considered algorithms takes no more than  $2n + y$  consecutive rounds to transmit successfully at least  $y$  packets. It takes at most  $n$  rounds with no transmission to identify stations that store at least  $y \leq x$  packets for SRR, because the token needs to make at most one full cycle along the list of stations. Similarly, it takes at most  $2n$  rounds with no transmission to identify stations that store at least  $y \leq x$  packets for OF-SRR. This is because two phases are needed in the worst case, when the first phase results in discovering too few old packets. Uploading these  $y$  packets takes  $y$  rounds. This completes the first part of the argument.

Next, we prove that these algorithms need no more than  $y \lg n$  consecutive rounds to upload  $y$  packets. Indeed, after a packet has been transmitted, it takes time at most the height of a conceptual search tree to identify another station with an eligible packet. This height is at most  $\lg n$ , so a packet is heard at least once in any contiguous segment of  $\lg n$  rounds.  $\square$

**Theorem 6** *The packet latency for algorithm OF-SRR executed against the adversary of type  $(\rho, b)$  is at most  $4 \min(b \lg n, n + b)$  for  $\rho \leq \frac{1}{2 \lg n}$ , and it is at most  $\frac{4n+2b}{1-\rho}$  for  $\rho > \frac{1}{2 \lg n}$ .*

**Proof:** A packet is transmitted successfully by the end of the phase following the one in which it was injected. So the maximum packet latency is upper bounded by twice the maximum length of a phase. At most  $\rho|\tau| + b$  packets get injected in a time interval  $\tau$  of length  $|\tau|$ . The maximum number of packets in the system is upper bounded by the maximum number of packets at the end of an interval plus the number of packets injected within this interval.

Let us begin with the case  $\rho \leq \frac{1}{2 \lg n}$ . We first argue that at the end of a round there are at most  $\rho \cdot \min(2b \lg n, 2n + 2b) + b$  packets in the system. Suppose it is otherwise and consider the first round  $t$  with the number of packets more than  $\rho \cdot \min(2b \lg n, 2n + 2b) + b$  at the end of it. Note that  $t > \min(2b \lg n, 2n + 2b)$ , as at most  $\rho \min(2b \lg n, 2n + 2b) + b$  packets are injected into the system in the time interval  $[1, \min(2b \lg n, 2n + 2b)]$ . This follows from the assumption that, at the end of round  $t - \min(2b \lg n, 2n + 2b)$ , there were at most

$$\rho \cdot \min(2b \lg n, 2n + 2b) + b$$

packets in the system. From that round until round  $t$ , at least this number of packets have been successfully transmitted, by Lemma 5 and the estimate

$$\rho \cdot \min(2b \lg n, 2n + 2b) + b \leq \frac{2b \lg n}{2 \lg n} + b = 2b ,$$

while at most  $\rho \cdot \min(2b \lg n, 2n + 2b) + b$  new packets have been injected. Consequently, the number of packets in the system at the end of round  $t$  would be at most  $\rho \cdot \min(2b \lg n, 2n + 2b) + b$ , which is a contradiction. This completes the proof that the number of packets in the system is at most  $\rho \cdot \min(2b \lg n, 2n + 2b) + b$ . This bound is also at most  $2b$ . By Lemma 5, a phase does not take longer than  $\min(2b \lg n, 2n + 2b)$  rounds, as the number of old packets in the system never surpasses  $2b$ . Hence, the packet latency is at most  $2 \cdot \min(2b \lg n, 2n + 2b) \leq 4 \min(b \lg n, n + b)$ .

The next case is when  $\rho > \frac{1}{2 \lg n}$ . Let  $x_i$  be the number of packets in the system in the beginning of phase  $i$ , and let  $t_i$  be the length of this phase. By Lemma 5, the inequalities  $x_1 \leq b$  and  $t_i \leq \min(x_i \lg n, 2n + x_i)$  hold, for every  $i \leq 1$ . This implies the inequality

$$x_{i+1} \leq \rho t_i + b \leq \rho \min(x_i \lg n, 2n + x_i) + b .$$

Iterating this formula yields

$$\begin{aligned} x_{i+1} &\leq b + \rho \min(x_i \lg n, 2n + x_i) \\ &\leq b + \rho (2n + (b + \rho \min(x_{i-1} \lg n, 2n + x_{i-1}))) \\ &= b + \rho (2n + b) + \rho^2 \min(x_{i-1} \lg n, 2n + x_{i-1}) , \end{aligned}$$

which in turn leads to the following inequalities:

$$\begin{aligned} x_{i+1} &\leq b + (2n + b)(\rho + \rho^2 + \dots + \rho^{i-1}) \\ &\leq \frac{\rho}{1-\rho} \left( 2n + \frac{b}{\rho} \right) = \frac{2n\rho + b}{1-\rho} . \end{aligned}$$

The packet latency is the maximum of  $t_i + t_{i+1}$ , over all  $i$ . By Lemma 5, these numbers are at most

$$\min(x_i \lg n, 2n + x_i) + \min(x_{i+1} \lg n, 2n + x_{i+1}) .$$

Each of them is upper-bounded by

$$4n + 2 \cdot \frac{2n\rho + b}{1 - \rho} \leq \frac{4n + 2b}{1 - \rho} ,$$

which completes the proof.  $\square$

**Theorem 7** *The packet latency of algorithm SRR executed against the adversary of type  $(\rho, b)$  is at most  $6b \lg n$  for  $\rho \leq \frac{1}{2 \lg n}$ , and at most  $\frac{4(n+b)}{(1-\rho)^2}$  for  $\rho > \frac{1}{2 \lg n}$ .*

**Proof:** Let  $x_i$  be the number of packets in the system in the beginning of phase  $i$ , and let  $t_i$  be the length of this phase.

We first consider the case of  $\rho \leq \frac{1}{2 \lg n}$ . By a similar argument as in the proof of Theorem 6, the number of pending packets is never more than

$$\rho \cdot \min(2b \lg n, 2n + 2b) + b \leq 2b .$$

This argument relies only on Lemma 5. Next, observe that the following inequality holds:

$$t_i \geq x_i + \rho t_i + b \geq \frac{t_i}{\lg n} .$$

This is because the number of all packets in the system in phase  $i$  multiplied by the maximum time  $\lg n$  of a void period in phase  $i$  is an upper bound on  $t_i$ . There are at most  $\rho t_i + b$  newly arrived packets in this phase. Combine this with the estimate  $\frac{t_i}{2 \lg n} \geq \rho t_i$  to obtain  $(x_i + b) \lg n \geq t_i$ . By an upper bound  $2b$  on the number of packets in the system, in the case of  $\rho \leq \frac{1}{2 \lg n}$ , we obtain that the following bound hold:

$$t_i \leq (x_i + b) \lg n \leq 3b \lg n .$$

Every packet arriving in phase  $i$  is transmitted by the end of phase  $i + 1$ . Hence the maximum packet latency is at most

$$\max_i (t_i + t_{i+1}) \leq 2 \cdot 3b \lg n \leq 6b \lg n .$$

In order to estimate packet latency in the case of  $\rho > \frac{1}{2 \lg n}$ , observe that the following inequalities hold:

$$t_i \geq x_i + (\rho t_i + b) \geq t_i - n .$$

This is because the number of all packets that are in the system in phase  $i$  plus the maximum total number  $n$  of void rounds in phase  $i$  is an upper bound on  $t_i$ . There are at most  $\rho t_i + b$  newly arrived packets in this phase, so  $\frac{x_i + n + b}{1 - \rho} \geq t_i$ . Lemma 5 for algorithm SRR implies that  $\frac{2\rho n + b}{1 - \rho}$  is an upper bound on the number of packets in the system in a round. It follows that the following bounds hold:

$$\begin{aligned} t_i &\leq \frac{x_i + n + b}{1 - \rho} \\ &\leq \frac{n(1 + \rho) + b(2 - \rho)}{(1 - \rho)^2} . \end{aligned}$$

Algorithm	Packet latency
OF-JRRW( $J$ )	$\mathcal{O}(\frac{bn}{(1-\lambda)(1-\rho-\lambda)})$
JRRW( $J$ )	$\mathcal{O}(\frac{bn}{(1-\lambda)(1-\rho-\lambda)^2})$
OFC-RRW	$\mathcal{O}(\frac{n+b}{1-\rho-\lambda})$
C-RRW	$\mathcal{O}(\frac{n+b}{(1-\rho-\lambda)^2})$
MBTF	$\frac{3n(n+b)}{(1-\lambda)(1-\rho-\lambda)} + \mathcal{O}(\frac{bn}{1-\lambda})$

Table 1: Upper bounds on packet latency for a channel with  $n$  stations with jamming when the injection rate  $\rho \geq 0$  and jamming rate  $\lambda \geq 0$  satisfy  $\rho + \lambda < 1$ . The inequality  $\lambda \leq J$  is assumed for algorithms OF-JRRW( $J$ ) and JRRW( $J$ ). Algorithms OF-JRRW( $J$ ) and JRRW( $J$ ) are non-adaptive, and the remaining algorithms are adaptive.

A packet arriving in phase  $i$  is transmitted by the end of phase  $i + 1$ , hence the maximum packet latency is upper bounded by

$$\begin{aligned} \max_i(t_i + t_{i+1}) &\leq 2 \cdot \frac{n(1 + \rho) + b(2 - \rho)}{(1 - \rho)^2} \\ &\leq \frac{4(n + b)}{(1 - \rho)^2}, \end{aligned}$$

which completes the proof.  $\square$

**Algorithm Move-Big-To-Front.** We conclude with packet latency of algorithm MOVE-BIG-TO-FRONT (MBTF), which is an adaptive algorithm for channels without collision detection. This algorithm was originally designed for channels without jamming, but using control bits in messages without packets allows to transfer the token without unnecessary delay even when a channel is subject to jamming. We may consider any of these two versions of the algorithm when there is no jamming in a channel. A translation of the bound on packet latency applies, in a manner similar to the case of full sensing algorithms for channels without collision detection.

**Corollary 3** *There are at most  $\frac{2\rho n(n+b)}{1-\rho} + \mathcal{O}(bn)$  packets in queues in any round when algorithm MBTF is executed against the adversary of type  $(\rho, b)$ .*

**Proof:** Follows from Lemma 3 when  $\lambda = 0$ .  $\square$

**Corollary 4** *The packet latency of algorithm MBTF is  $\mathcal{O}(\frac{n(n+b)}{1-\rho})$  when executed against the adversary of type  $(\rho, b)$ .*

**Proof:** The bound on packet latency given in Theorem 5 becomes  $\mathcal{O}(\frac{n(n+b)}{1-\rho})$  for  $\lambda = 0$ .  $\square$



Algorithm	Packet latency	Injection rate
OF-RRW	$\mathcal{O}\left(\frac{n+b}{1-\rho}\right)$	$\rho < 1$
RRW	$\mathcal{O}\left(\frac{n+b}{(1-\rho)^2}\right)$	$\rho < 1$
OF-SRR	$4 \min(b \lg n, n + b)$	$\rho \leq \frac{1}{2 \lg n}$
SRR	$6b \lg n$	$\rho \leq \frac{1}{2 \lg n}$
OF-SRR	$\frac{4n+2b}{1-\rho}$	$\frac{1}{2 \lg n} < \rho < 1$
SRR	$\frac{4(n+b)}{(1-\rho)^2}$	$\frac{1}{2 \lg n} < \rho < 1$
MBTF	$\mathcal{O}\left(\frac{n(n+b)}{1-\rho}\right)$	$\rho < 1$

Table 2: Upper bounds on packet latency for a channel with  $n$  stations without jamming, depending on injection rate  $\rho \geq 0$ . Algorithm MBTF is adaptive, and the remaining algorithms are non-adaptive.

## 7 Conclusion

We study worst-case packet latency of deterministic distributed broadcast algorithms in adversarial multiple-access channels, in which an adversary controls packet injection and, optionally, jamming. We derived asymptotic upper bounds on packet latency expressed in terms of the quantitative constraints defining adversaries. The upper bounds on packet latency are summarized in Tables 1 and 2.

When the channel is with jamming, then there is a non-adaptive algorithm with bounded packet latency for any injection rate smaller than 1, but this algorithm operates only for a known range of values of jamming rates. It is an open problem if there is a non-adaptive algorithm that is stable against all jamming adversaries with the injection rate  $\rho$  and jamming rate  $\lambda$  satisfying  $\rho + \lambda < 1$ , as already discussed at the end of Section 4.

When the channel is without jamming, then algorithms OF-SRR and SRR have packet latencies that are  $o(n)$ , for suitably small injection rates as functions of  $n$ , when burstiness is treated as a constant. It is unknown if such algorithms exist for channels with jamming.

Algorithm MBTF is the only one we discussed with the property that its packet latency is  $\Theta(n^2)$ , when burstiness  $b$  and injection rate  $\rho$  and jamming rate  $\lambda$  are all treated as constants, and when  $\rho + \lambda < 1$ . It is also the only algorithm among those we considered for which there is an upper bound on the number of packets queued in the system, which is independent of injection and jamming rates, because this bound holds even for  $\rho + \lambda = 1$ , see [23]. It is unknown if there is an algorithm stable when  $\rho + \lambda = 1$  and such that its packet latency is  $o(n^2)$  when  $\rho + \lambda < 1$  and when burstiness  $b$  and injection rate  $\rho$  and jamming rate  $\lambda$  are all treated as constants.

This paper considers performance of deterministic distributed broadcast protocols in terms of their worst-case packet latency. Although theoretically appealing, this measure of performance could be considered as less relevant to real-world applications than average packet latency. Proposing adversarial models for packet injection to study average packet latency of deterministic algo-

rithms would be an interesting direction of future work.

The model of channels we consider is idealized and simplified to a considerable extent, in particular, we assume that the set of stations attached to the channel is fixed and time is slotted into rounds. It is an interesting area of future work to consider deterministic broadcasting algorithms in less restricted models of multiple access channels; paper [5] indicates one such a possible direction.

## References

- [1] W. Aiello, E. Kushilevitz, R. Ostrovsky, and A. Rosén. Adaptive packet routing for bursty adversarial traffic. *Journal of Computer and System Sciences*, 60(3):482–509, 2000.
- [2] H. Al-Ammal, L. A. Goldberg, and P. D. MacKenzie. An improved stability bound for binary exponential backoff. *Theory of Computing Systems*, 34(3):229–244, 2001.
- [3] C. Àlvarez, M. J. Blesa, J. Díaz, M. J. Serna, and A. Fernández. Adversarial models for priority-based networks. *Networks*, 45(1):23–35, 2005.
- [4] C. Àlvarez, M. J. Blesa, and M. J. Serna. The impact of failure management on the stability of communication networks. In *Proceedings of the 10th International Conference on Parallel and Distributed Systems (ICPADS)*, pages 153–160, 2004.
- [5] L. Anantharamu and B. S. Chlebus. Broadcasting in ad hoc multiple access channels. *Theoretical Computer Science*, 584:155–176, 2015.
- [6] L. Anantharamu, B. S. Chlebus, D. R. Kowalski, and M. A. Rokicki. Deterministic broadcast on multiple access channels. In *Proceedings of the 29th IEEE International Conference on Computer Communications (INFOCOM)*, pages 1–5, 2010.
- [7] L. Anantharamu, B. S. Chlebus, D. R. Kowalski, and M. A. Rokicki. Medium access control for adversarial channels with jamming. In *Proceedings of the 18th International Colloquium on Structural Information and Communication Complexity (SIROCCO)*, Lecture Notes in Computer Science, vol. 6796, pages 89–100. Springer, 2011.
- [8] L. Anantharamu, B. S. Chlebus, and M. A. Rokicki. Adversarial multiple access channels with individual injection rates. *Theory of Computing Systems*, 2017. Published online in 2016 at doi:10.1007/s00224-016-9725-x.
- [9] M. Andrews, B. Awerbuch, A. Fernández, F. T. Leighton, Z. Liu, and J. M. Kleinberg. Universal-stability results and performance bounds for greedy contention-resolution protocols. *Journal of the ACM*, 48(1):39–69, 2001.
- [10] M. Andrews, A. Fernández, A. Goel, and L. Zhang. Source routing and scheduling in packet networks. *Journal of the ACM*, 52(4):582–601, 2005.
- [11] M. Andrews and L. Zhang. Achieving stability in networks of input-queued switches. *IEEE/ACM Transactions on Networking*, 11(5):848–857, 2003.
- [12] M. Andrews and L. Zhang. Routing and scheduling in multihop wireless networks with time-varying channels. *ACM Transactions on Algorithms*, 3(3):33, 2007.

- [13] A. F. Anta, M. A. Mosteiro, and J. R. Muñoz. Unbounded contention resolution in multiple-access channels. *Algorithmica*, 67(3):295–314, 2013.
- [14] B. Awerbuch, A. W. Richa, and C. Scheideler. A jamming-resistant MAC protocol for single-hop wireless networks. In *Proceedings of the 27th ACM Symposium on Principles of Distributed Computing (PODC)*, pages 45–54, 2008.
- [15] M. A. Bender, M. Farach-Colton, S. He, B. C. Kuszmaul, and C. E. Leiserson. Adversarial contention resolution for simple channels. In *Proceedings of the 17th Annual ACM Symposium on Parallel Algorithms (SPAA)*, pages 325–332, 2005.
- [16] M. A. Bender, J. T. Fineman, S. Gilbert, and M. Young. How to scale exponential backoff: Constant throughput, polylog access attempts, and robustness. In *Proceedings of the 27th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 636–654, 2016.
- [17] M. A. Bender, T. Kopelowitz, S. Pettie, and M. Young. Contention resolution with log-logstar channel accesses. In *Proceedings of the 48th ACM Symposium on Theory of Computing (STOC)*, pages 499–508, 2016.
- [18] M. Bieńkowski, T. Jurdziński, M. Korzeniowski, and D. R. Kowalski. Distributed online and stochastic queuing on a multiple access channel. In *Proceeding of the 26th International Symposium on Distributed Computing (DISC)*, volume 7611 of *Lecture Notes in Computer Science*, pages 121–135. Springer, 2012.
- [19] A. Borodin, J. M. Kleinberg, P. Raghavan, M. Sudan, and D. P. Williamson. Adversarial queuing theory. *Journal of the ACM*, 48(1):13–38, 2001.
- [20] A. Z. Broder, A. M. Frieze, and E. Upfal. A general approach to dynamic packet routing with bounded buffers. *Journal of the ACM*, 48(2):324–349, 2001.
- [21] B. S. Chlebus. Randomized communication in radio networks. In P. M. Pardalos, S. Rajasekaran, J. H. Reif, and J. D. P. Rolim, editors, *Handbook of Randomized Computing*, volume I, pages 401–456. Kluwer Academic Publishers, 2001.
- [22] B. S. Chlebus, G. De Marco, and D. R. Kowalski. Scalable wake-up of multi-channel single-hop radio networks. *Theoretical Computer Science*, 615:23–44, 2016.
- [23] B. S. Chlebus, D. R. Kowalski, and M. A. Rokicki. Maximum throughput of multiple access channels in adversarial environments. *Distributed Computing*, 22(2):93–116, 2009.
- [24] B. S. Chlebus, D. R. Kowalski, and M. A. Rokicki. Adversarial queuing on the multiple access channel. *ACM Transactions on Algorithms*, 8(1):5:1–5:31, 2012.
- [25] G. De Marco and D. R. Kowalski. Fast nonadaptive deterministic algorithm for conflict resolution in a dynamic multiple-access channel. *SIAM Journal of Computing*, 44(3):868–888, 2015.
- [26] S. Gilbert, R. Guerraoui, D. R. Kowalski, and C. Newport. Interference-resilient information exchange. In *Proceedings of the 28th IEEE International Conference on Computer Communications (INFOCOM)*, pages 2249–2257, 2009.

- [27] S. Gilbert, R. Guerraoui, and C. C. Newport. Of malicious notes and suspicious sensors: On the efficiency of malicious interference in wireless networks. *Theoretical Computer Science*, 410(6-7):546–569, 2009.
- [28] S. Gilbert, V. King, S. Pettie, E. Porat, J. Saia, and M. Young. (Near) optimal resource-competitive broadcast with jamming. In *Proceedings of the 26th ACM Symposium on Parallelism in Algorithms and Architectures (SPAA)*, pages 257–266, 2014.
- [29] L. A. Goldberg, M. Jerrum, S. Kannan, and M. Paterson. A bound on the capacity of backoff and acknowledgment-based protocols. *SIAM Journal on Computing*, 33(2):313–331, 2004.
- [30] L. A. Goldberg, P. D. MacKenzie, M. Paterson, and A. Srinivasan. Contention resolution with constant expected delay. *Journal of the ACM*, 47(6):1048–1096, 2000.
- [31] J. Håstad, F. T. Leighton, and B. Rogoff. Analysis of backoff protocols for multiple access channels. *SIAM Journal on Computing*, 25(4):740–774, 1996.
- [32] D. R. Kowalski. On selection problem in radio networks. In *Proceedings of the 24th ACM Symposium on Principles of Distributed Computing (PODC)*, pages 158–166, 2005.
- [33] D. Meier, Y. A. Pignolet, S. Schmid, and R. Wattenhofer. Speed dating despite jammers. In *Proceedings of the 5th IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS)*, Lecture Notes in Computer Science vol. 5516, pages 1–14, 2009.
- [34] P. Raghavan and E. Upfal. Stochastic contention resolution with short delays. *SIAM Journal on Computing*, 28(2):709–719, 1998.
- [35] A. W. Richa, C. Scheideler, S. Schmid, and J. Zhang. Competitive throughput in multi-hop wireless networks despite adaptive jamming. *Distributed Computing*, 26(3):159–171, 2013.
- [36] A. Rosén and M. S. Tsirkin. On delivery times in packet networks under adversarial traffic. *Theory of Computing Systems*, 39(6):805–827, 2006.
- [37] C. Scheideler and B. Vöcking. Universal continuous routing strategies. *Theory of Computing Systems*, 31(4):425–449, 1998.